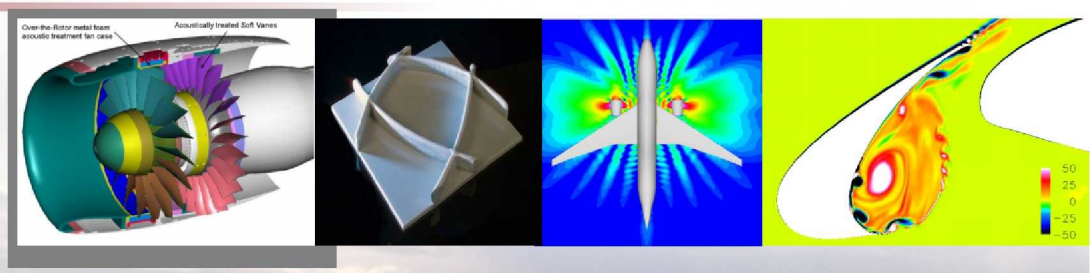
OpenMDAO Framework Status

Advancing and exploring the science of Multidisciplinary Analysis & Optimization (MDAO) capabilities are high-level goals in the Fundamental Aeronautics Program's Subsonic Fixed Wing (SFW) project. The OpenMDAO team has made significant progress toward completing the Alpha OpenMDAO deliverable due in September 2010. Included in the presentation are: details of progress on developing the OpenMDAO framework, example usage of OpenMDAO, technology transfer plans, near term plans, progress toward establishing partnerships with external parties, and discussion of additional potential collaborations.

OpenMDAO Framework Status

Cynthia Gutierrez Naiman
NASA



48th AIAA Aerospace Sciences Meeting
Orlando, FL
January 4, 2010

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Background

- **What is our goal?**
  - Advance Physics-Based MDAO Capabilities to Address Gaps

- **Why?**
  - Physics-Based MDAO is critical in designing & optimizing unconventional configurations, which are essential to reduce noise, emissions, & fuel burn, while increasing performance.
    - Directly supports principles in National Aeronautics Research & Development Policy (Dec 2006) http://www.aeronautics.nasa.gov/releases/national_aeronautics_rd_policy_dec_2006.pdf
    - Addresses challenges in National Plan for Aeronautics R&D & Related Infrastructure (Dec 2007) http://www.aeronautics.nasa.gov/releases/aero_rd_plan_final_21_dec_2007.pdf

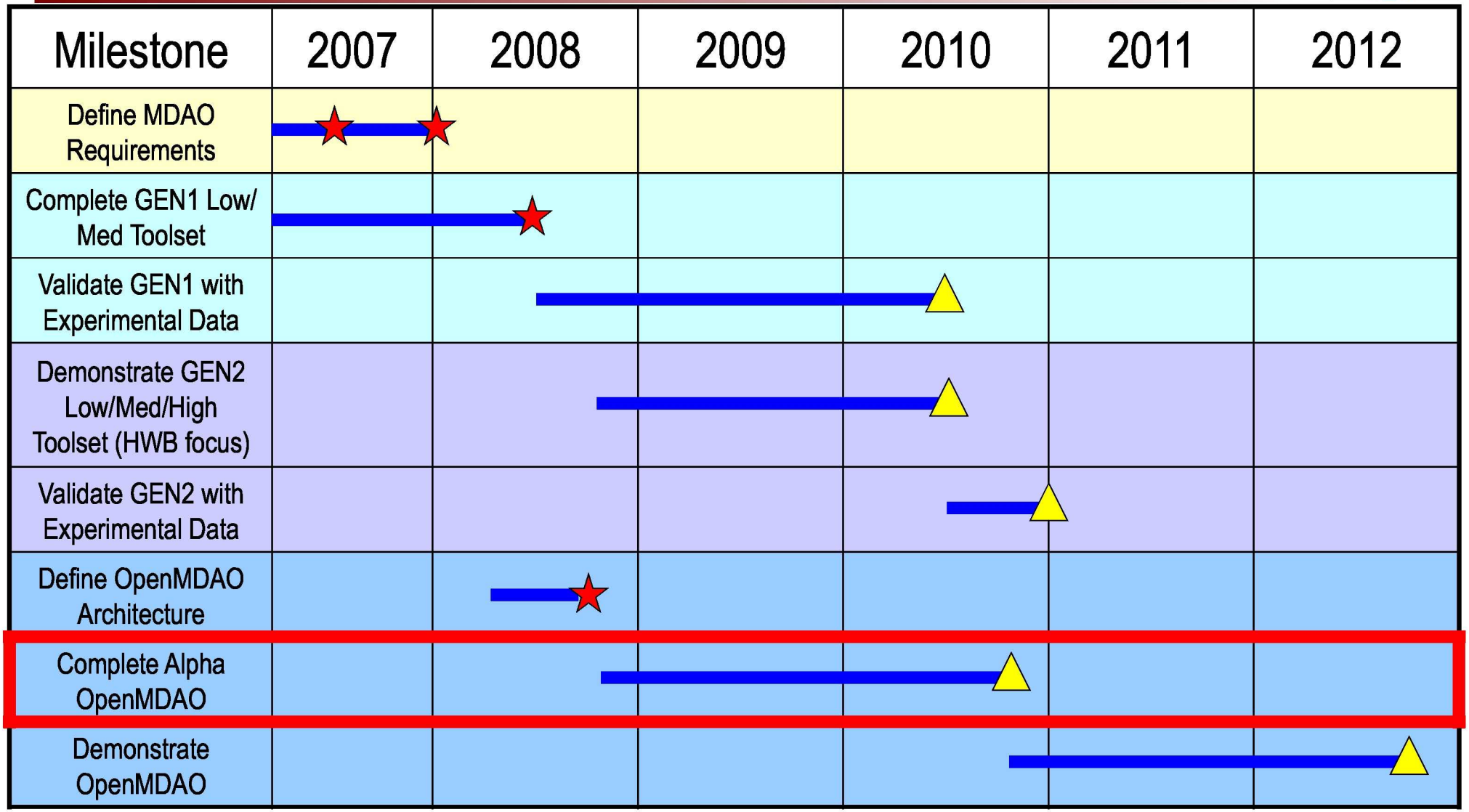
- **MDAO Task**
  - Near-Term Path
    - Improve current tools to meet immediate application-focused needs & milestones
  - Far-Term Path
    - Develop an open source Next Generation MDAO environment to reach the broadest possible audience
    - Advance & explore the science of MDAO

- **Stakeholders/Customers: MDAO community**

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Milestones



Completed Milestone

Timeline

Planned Milestone

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

OpenMDAO Team Acknowledgement

Name	Team Role
Bret Naylor	Senior Software Developer
Scott Townsend	Senior Software Developer
Ken Moore	Engineer/Software Developer
Justin Gray	Engineer/Software Developer
Keith Marsteller	Software Developer
Cathy Krenek	V&V Lead
Rula Coroneos	Developer/Tester
Suresh Khandelwal	Developer/Tester
Tina Grzincic	Software Configuration Manager
Paulette Ziegfeld	Technical Writer
James Below	System Administrator

Major Features of MDAO Vision

- [INFRASTRUCTURE] Computational infrastructure for analysis integration
- [MODELING TOOLS] Component disciplinary analyses and attendant tools
- [REQUIREMENTS FORMULATION] System/design/requirements formulation
- [PROBLEM DESCRIPTION] Problem component description
- [PROBLEM FORMULATION] Design problem formulation
- [PROBLEM SOLUTION] Design problem solution

What is OpenMDAO?

- Computational Environment (Framework)
  - Will allow integration of multidiscipline codes of multiple fidelities (low, medium, high) into an engineering process
    - Open source framework does not require that components be open source.
    - Analysis codes, optimization algorithms, etc., can be proprietary, secret, etc.
    - There are requirements to restrict access to components, process models, etc., based on user class.
  - Will include drivers useful for design optimization – gradient and genetic optimizers, sensitivity analysis, design of experiments, Pareto analysis, etc.
  - Is application neutral (i.e., not restricted to aerospace problems)
- Open Source
  - User is free to download, use, modify, and distribute
  - Fixes and enhancements are contributed by community
- Community
  - Forum for fostering discussion amongst the MDAO community
  - Hosting for plugins and components contributed by the public

OpenMDAO Status: Implementation (1 of 7)

- Implemented Core Framework Classes: a basic functional framework to allow model building
  - Components
    - Basic building block of an analysis
  - Assemblies
    - Amalgamation of components
    - Top level of model is an assembly
  - Component linking
  - Data passing
    - Variables types, unit conversions
  - Sockets & Interfaces (from Traits)
  - Driver interface
    - Special component that coordinates execution of other components
    - Optimizers, solvers, iterators are drivers
      - CONMIN (gradient optimizer)
      - PyEvolve (genetic optimizer)
      - Case Iterator
  - Data Flow & Lazy Evaluation
    - Component execution order inferred by Data Flow
      - Components only execute if inputs become invalid
        - » Otherwise known as Lazy Evaluation
      - Some data flows will be parallelizable
    - Graph data structures and algorithms implemented using Python package networkx (LGPL)
    - Drivers operate on their sibling components
      - Every assembly has a workflow with some number of drivers (n>=0)
      - Eliminating excessive nesting of assemblies was a design decision
        - » This may need to change when explicit workflows are implemented
    - More complicated data flows with nested drivers have been tested

OpenMDAO Status: Implementation (2 of 7)

- Implemented framework variable types using Enthought's Traits
  - Enthought's Traits:
    - Strong typing to Python
    - Validation of attributes (including conversion)
    - Initialization – setting and restoring defaults
    - Notification – function callback on set
    - Delegation & visualization (currently unused)
  - Standard types available – int, float, string, arrays
  - New data types can be created
    - StringRef – points to a variable in the model hierarchy
    - UnitsFloat – a float with units
  - Unit checking & conversion between output and inputs

- Buildout: using **zc.buildout** to assure a common environment for OpenMDAO users and developers
  - Creates a clean isolated Python environment
  - Automates installation of required eggs and their dependencies
  - Uses recipes to create custom configurations for tools

OpenMDAO Status: Implementation (3 of 7)

- Implemented two Save & Load scenarios
  - Three basic scenarios:
    - Save state for reloading – completed
    - Save entire configuration for transport (to another user, concurrent evaluation, etc.) – completed
    - Save checkpoint for later restart (includes internal state & modified external files) – not implemented
  - Save files are "eggs," zip files with standard metadata format for package dependencies, versioning, etc.

- Created tool to take a single Python module containing an OpenMDAO plugin definition and turn it into an egg with appropriate entry points that will allow auto-discovery of the plugin by the framework.

- Integrated pyNPSS into OpenMDAO framework

- Updated to Python 2.6

OpenMDAO Status: Implementation (4 of 7)

Current Third-party Eggs:

Mathematical
conmin – Gradient optimizer
numpy – General numerical and mathematical package, arrays
PyEvolve – Genetic algorithms
scipy – Science/Math, numerical integration
ScientificPython – Unit definitions

Infrastructure
networkx – network graph algorithms
pyparsing – statement parsing
PyYAML – object serialization
setuptools – component packaging
zc.buildout – isolated Python environment
ZopeSkel - namespaces

Documentation
Jinja2 – templating library
PIL (Python Image Library) – general image processing library
Pygments – code highlighting
Sphinx – converts restructured text (reST) → html

Testing
coverage – reports how much code is covered by the test suite
docutils – generates documentation from docstrings in code
nose – test harness
pylint – verifies that code follows standards

OpenMDAO Status: Implementation (5 of 7)

Examples using OpenMDAO foundation classes:

- Ported a supersonic engine model employing five instances of NPSS
  - Simulation incorporates multiple OpenMDAO NPSS component instances running in separate directories communicating via linked scalar and file variables to generate data for FLOPS, ANOPP, etc., from high-level design inputs.
- Wrapped M4 Engineering's Design Of Experiments (DOE) and mid-fidelity codes as an OpenMDAO driver and component, respectively
  - Codes are from M4 Engineering's Python Multidisciplinary Optimization Object Library (MOOL).
  - DOE driver is based on the OpenMDAO case iterator driver, which is designed to support concurrent evaluation of cases.
  - Mid-fidelity component is an example of an OpenMDAO component using sockets to run other wrapped components.
- Ported a two-level genetic optimization of a variable-cycle NPSS engine model
  - Simulation uses nested instances of OpenMDAO genetic optimizers and the OpenMDAO NPSS component to determine the best parameters for design and off-design performance.

OpenMDAO Status: Implementation (6 of 7)

- Developing User Interface
  - Planning to implement a browser-based GUI: users will interact with the GUI through a browser both locally and remotely
  - Designing toward having complete functionality in both GUI and console modes
  - Researching GUI preliminary design
    - Investigated web-based framework tools:
      - Pylons, Turbogears
      - Ruby on Rails
      - Django
    - Investigated GUI toolkit technologies:
      - Asynchronous Javascript And XML (AJAX)
      - JQuery library, JQuery UI
      - ExtJS
      - Pyjamas
  - Generating screen mock-ups
  - Interviewing potential users to solicit feedback

OpenMDAO Status: Implementation (7 of 7)

- Explored Python Multiprocessing
  - Multiprocessing is part of Python version 2.6.
  - Allows multiple CPUs to process a single Python program in a manner similar to threads.
  - Uses local machine. Remotes hosts possible but not officially supported.
  - Requires declaration of methods and data types on both client and server.
  - Connections authenticated, but data sent in the clear.

- Geometry Capability
  - OpenMDAO process integrates multiple disciplines based on a common geometry model into an optimization problem
  - Component-level and system-level APIs being developed
    - Manipulation of Parameters and Feature Suppression states
    - Query of the Geometry Object to extract information from the geometry (i.e., for meshing)
    - Interaction with the Mesh Objects by analysis tools
    - Additional API to support an open-source path for handling geometry
  - CAPRI being considered as part of the component-level API
    - Provides a common interface for most commonly-used CAD kernels

- Planning to support 3 platforms: Windows, Linux, Mac

OpenMDAO Status: Software Quality

- Adhere to coding standards
  - Use pylint to monitor coding standards PEP-8 compliance (style guide for Python code)
- Conduct code peer reviews & formal inspections
- Document software development process & collect metrics
- Set up the development environment (Bazaar, Trac)
- Incorporate verification testing
  - Current suite of unit tests includes 200 tests
  - Code coverage is 85%
- Undergo validation testing
  - M4 Engineering validated OpenMDAO framework functionality by successfully replicating High Speed Civil Transport analysis

OpenMDAO Status: User Documentation

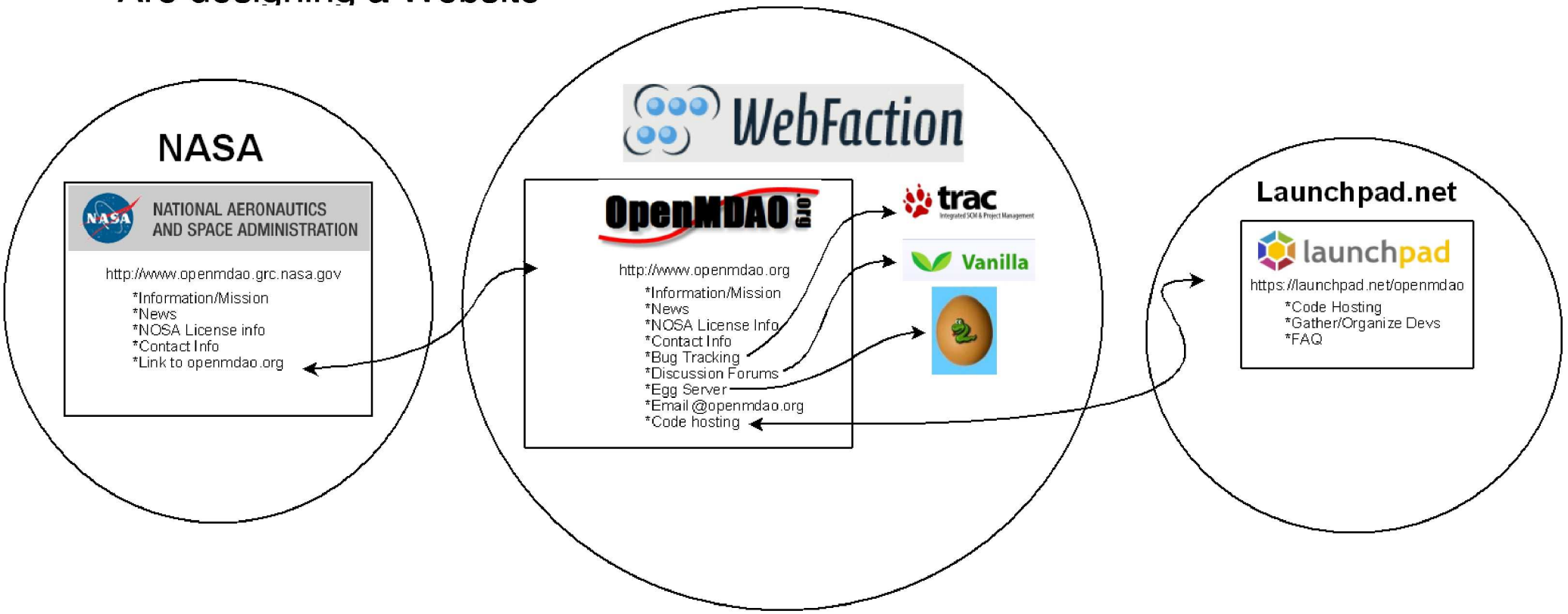
- Architecture Document
- User's Guide
  - Includes Tutorial Problem
  - Uses doctest plugin to Sphinx to run tests on code excerpts included in documentation
- Developer's Guide
- Plugin Developer's Guide
- Source Documentation
- Licenses

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Outreach: Technology Transfer Preparation

- Requested open source public domain dissemination approval
- Researched what's needed to host an open source project
- Are designing a Website



This diagram represents an overview of the planned OpenMDAO Web Presence, including a NASA Website, an OpenMDAO.org community Website hosted by WebFaction, and a Launchpad.net code-hosting site.

Outreach to MDAO Community

- Government:
  - DoD
    - Computational Research & Engineering Acquisition Tools & Environments (CREATE) Program: hosted meeting & discussed architecture in detail
    - AFRL Air Vehicles & Propulsion Directorates: continue knowledge sharing on similar MDAO objectives
  - DoE
    - Sandia Labs: Design Analysis Kit for Optimization and Terascale Applications (DAKOTA): NASA currently drafting Inter-Agency Agreement.

- Industry:
  - Invited to brief OpenMDAO status to Boeing enterprise MDAO representatives. Establishing Space Act Agreement with Boeing
  - Held initial telecoms with Northrop Grumman & GE Aviation to discuss potential MDAO collaboration
  - Contacted NPSS partners to see if interested in MDAO activity. Potential partners who have expressed interest are: Lockheed Martin, Williams International, Rolls-Royce, and Honeywell

- Academia:
  - Georgia Tech: agreed to be Beta Tester
  - Held brief discussions on potential interest with MIT, Stanford, University of Toronto

Goal: Parties to agreements will be the external partner and the four NASA Aero Centers (Glenn, Langley, Dryden, Ames).

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Near Term Plans (1 of 2)

- Technical
  - Update documentation
  - Set up development environment on Mac hardware
  - Add ability to explicitly specify workflow in an Assembly
  - Add distributed execution capability
  - Implement a database CaseRecorder for saving completed cases
  - Create a GUI prototype
  - Add drivers
  - Develop a library to support file wrapping
  - Create a Matlab component (possibly Octave as well)
  - Integrate Aero tools: target GEN2 codes used for HWB focus
  - Continue definition of Geometry API, e.g., query API and mesh object API

Near Term Plans (2 of 2)

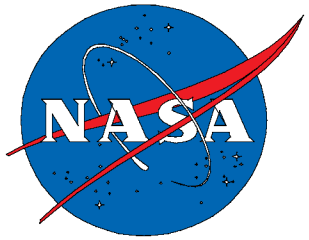
- Task Management
  - Complete OpenMDAO Framework software plans
  - Revisit discipline requirements with experts
  - Complete “open source public domain” approval process
  - Continue reaching out to potential partners/users
    - Establish agreements if needed; contact (Cynthia.G.Naiman@nasa.gov)
    - Work as a group
  - If approved, go live at

Topic Outline

- Background
- Milestones
- OpenMDAO Status
- Outreach
- Near-Term Plans
- Conclusion

Conclusion

- Advancing Physics-Based MDAO is critical to supporting principles in National Aeronautics Research & Development Policy
- Developing an open source capability ensures the widest possible dissemination of information & knowledge sharing
- Involving users throughout the development lifecycle is critical
- Partnering with industry, academia, and other government agencies is essential to realize MDAO vision, which is to advance the science of MDAO


NASA

OpenMDAO Status: Requirements Management

OpenMDAO Framework Tickets Statistics

Internal Milestone	2009_09_(Fall)			2010_03_ (Spring)			2010_09_(Fall)			
Type →	REQ	ENH	DEF	REQ	ENH	DEF	REQ	ENH	DEF	Status Totals
Ticket Status ↓										
New	12	3	1	4	1		7	2	2	32
Accepted	1	1						1		3
Assigned	1	3								4
Working		1						1		2
Ready_Review	3	8	1		1					13
Merged	15	2	1							18
Ticket Type Totals	**32**	**18**	**3**	**4**	**2**	**0**	**7**	**4**	**2**	
Total Tickets	**53**			**6**			**13**			**72**

REQ – requirement
ENH – enhancement
DEF – defect

Backup Slides

Outreach: NRA and SBIR Activities

- **Subsonic Fixed Wing NRAs**
  - Develop parametric blade geometry modeler **(AVETeC/University of Cincinnati)**
  - "System Analysis & Design Approach to the Hybrid Wing/Body Aircraft" **(AVID LLC)**
  - Improve structural modeling, meshing and rapid grid morphing capabilities within Vehicle Sketch Pad **(Cal Poly/Phoenix Integration/J R Gouldemans)**
  - "Enhanced Modeling & Analysis for Emission Prediction" **(Georgia Tech/ASDL)**
  - "Adv Multidisciplinary Optimization Techniques for Efficient Subsonic Aircraft Design" **(MIT/ Stanford/Purdue/Boeing)**

- **Supersonics NRAs**
  - "High Fidelity MDO: Software Infrastructure & Application to Supersonic Aircraft" **(M4 Engineering/ Phoenix Integration)**
  - "Control of Boundary Representation Topology in MDAO" **(MIT)**
  - "Multifidelity Analysis and Design Methods for Supersonic Aircraft" **(Stanford/MIT)**

- **Subsonic Fixed Wing SBIRs**
  - "Integrated Multidisciplinary Optimization Objects" **(M4 Engineering)**
  - "Variable-Fidelity Conceptual Design System for Advanced Unconventional Air Vehicles" **(Continuum Dynamics, Inc.)**
  - "Integration of an Advanced Cryogenic Electric Propulsion System (ACEPS) to Aerodynamically Efficient Subsonic Transport Aircraft" **(Empirical Systems Aerospace, LLC)**
  - "A Physics-based Starting Model for Gas Turbine Engines" **(EcoPro Technologies, LLC)**
  - "Integrated Network of Optimizations for Aircraft Systems" **(Michigan Engineering Services, LLC)**

Details on any NRA & SBIR efforts are available.

Outreach: Establish Partnerships

- Identify & follow up with interested parties, establish agreements, and **work as a group**
- Systems Analysis Design & Optimization (SAD&O) Website:
  - http://mdao.grc.nasa.gov/SADO-TWG/index.html
    - Userid "guest"
    - Password "sado"
- Contact Bill Haller for SAD&O-specific information: William.J.Haller@nasa.gov 216-977-7004
- Contact Cynthia Naiman for MDAO-specific information: Cynthia.G.Naiman@nasa.gov 216-433-5238

Iterative Software Development Activities

Phase	Activities	Products	Status
Requirements Development	• Develop Software Plans • Elicit & Analyze Requirements • Write Specification • Validate Requirements • Perform Inspections & Reviews • Perform Testability & Traceability Analyses	• Vision & Scope (V&S) Document • Use Case Document • Software Requirements Specification (SRS) • Software Management Plan(s) • Requirements Traceability Matrix	• Completed V&S Doc, Use Case Doc, SRS (functional & non-functional reqs) Glossary • Completed DRAFTs of Software Management Plan, Software V&V Plan • Collected feedback on discipline requirements
Architecture & Design	• Define criteria for frameworks, identify & assess • Prototype framework capabilities • Demonstrate prototype(s) & incorporate feedback • Develop initial architecture, design, & interfaces • Perform Inspections & Reviews • Incorporate knowledge gained from partnerships • Experts take inventory of their codes & plan	• Framework Assessment • Prototypes • Architecture, Design, & Interface Document	• Completed framework assessment • Developed prototypes • Completed Architecture Doc
Implementation & Testing	• Develop & conduct tests to see if requirement is met • Implement capabilities • Develop/Enhance/Integrate Discipline Codes • Perform Inspections & Reviews • Distribute release	• Alpha Release (9/30/10) • Release (9/30/12) • Release Notes • User Documentation • Test Plans & Test Results	• Implemented core framework classes • Started user documentation • Validated core framework • Conducting peer reviews
Acceptance Testing	• Perform tests • Conduct review	• Validated MDAO Capability	

Open Source Option in Government

- FederalComputerWeek June 15, 2009, article on open source in the government: http://fcw.com/Articles/2009/06/08/feature-Open-source.aspx
  - "Vivek Kundra, the federal government's new chief information officer, includes open source as one of the technologies he supports using to make government work better and more cheaply…"
- June 1, 2009, Homeland Open Security Technology (HOST) program launched:
  - Collaborative venture between Department of Homeland Security Science & Technology Directorate, University of Southern Mississippi, & Department of the Navy Space and Naval Warfare Systems Command to promote the use of Open Technology Solutions (OTS) in government.

    http://www.oss-institute.org/index2.php?option=com_content&do_pdf=1&id=338
    - From HOST Overview:

      http://oss-institute.org/HOST09/HOST_master_brief_08102009_overview.pdf
  - Benefits of Open Technology Solutions (OTS):
    - Encourages Competition for Development, Service and Support of IT
    - Discourages "Vendor Lock-in"
    - Increases Technical Efficiency and Security through Access and Code Review by Users and Developers
- NASA Open Source Agreement (NOSA) is Open Source Initiative Certified

2004 NASA Open Source Agreement (NOSA)

- NASA Open Source Agreement (NOSA) is Open Source Initiative (OSI) Certified

- Ref: "Developing An Open Source Option for NASA Software" by Patrick J. Moran, NASA Ames Research Center, NAS Technical Report NAS-03-009, April 21, 2003

  For NASA, the adoption of an Open Source option for software distribution would lead to three main benefits:

  1. Improved software development

     "Open Source enables a type of peer review for software."

  2. Enhanced collaboration, in particular across organizational boundaries

     "NASA's overall mission … is not one that NASA can effectively achieve alone. To be successful, NASA will need to work with other government agencies, academia and industry. Open Source makes those types of collaborations easier."

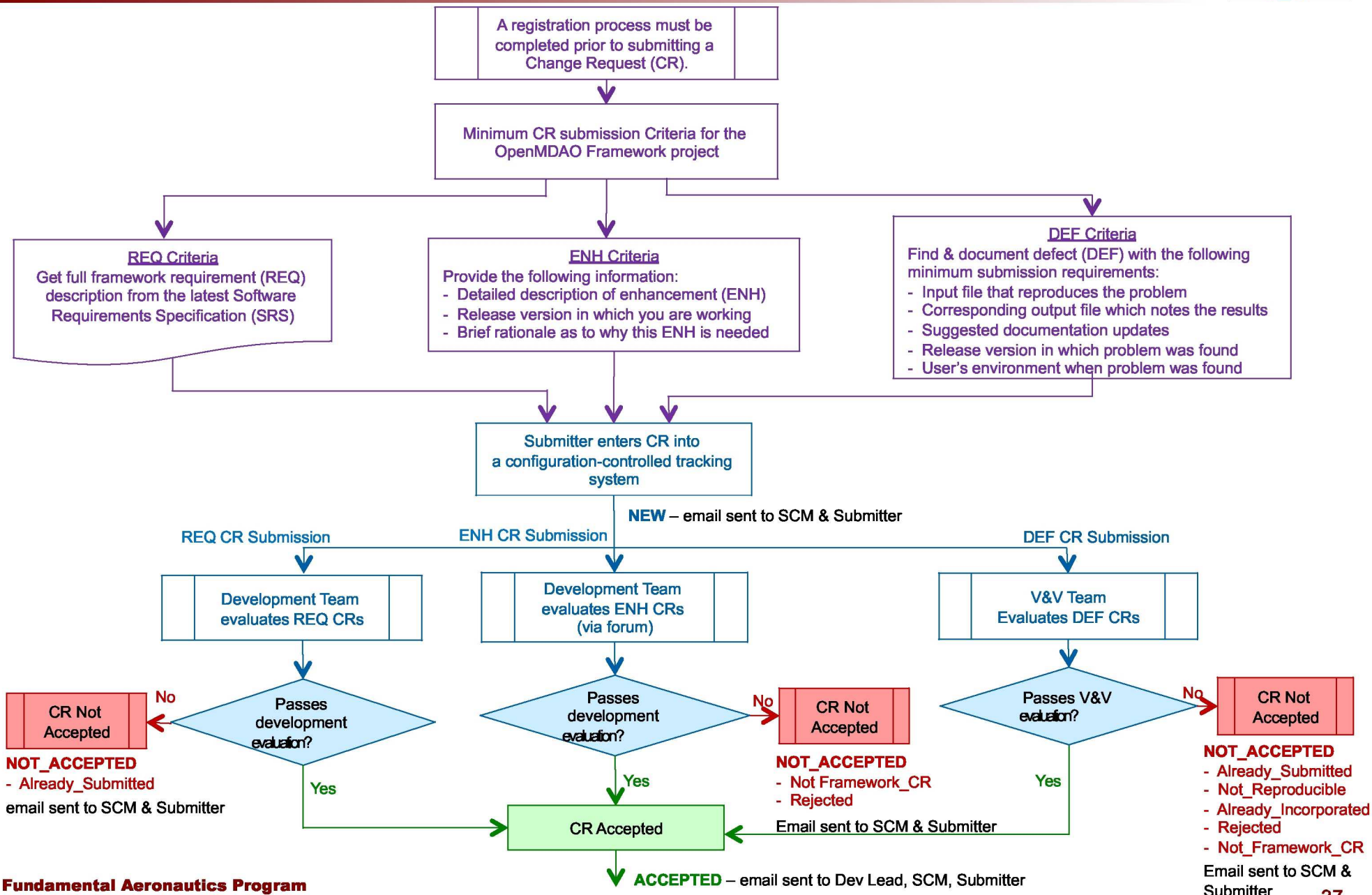
  3. More efficient and effective dissemination

     "… from the beginning NASA has been directed to 'provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof' "

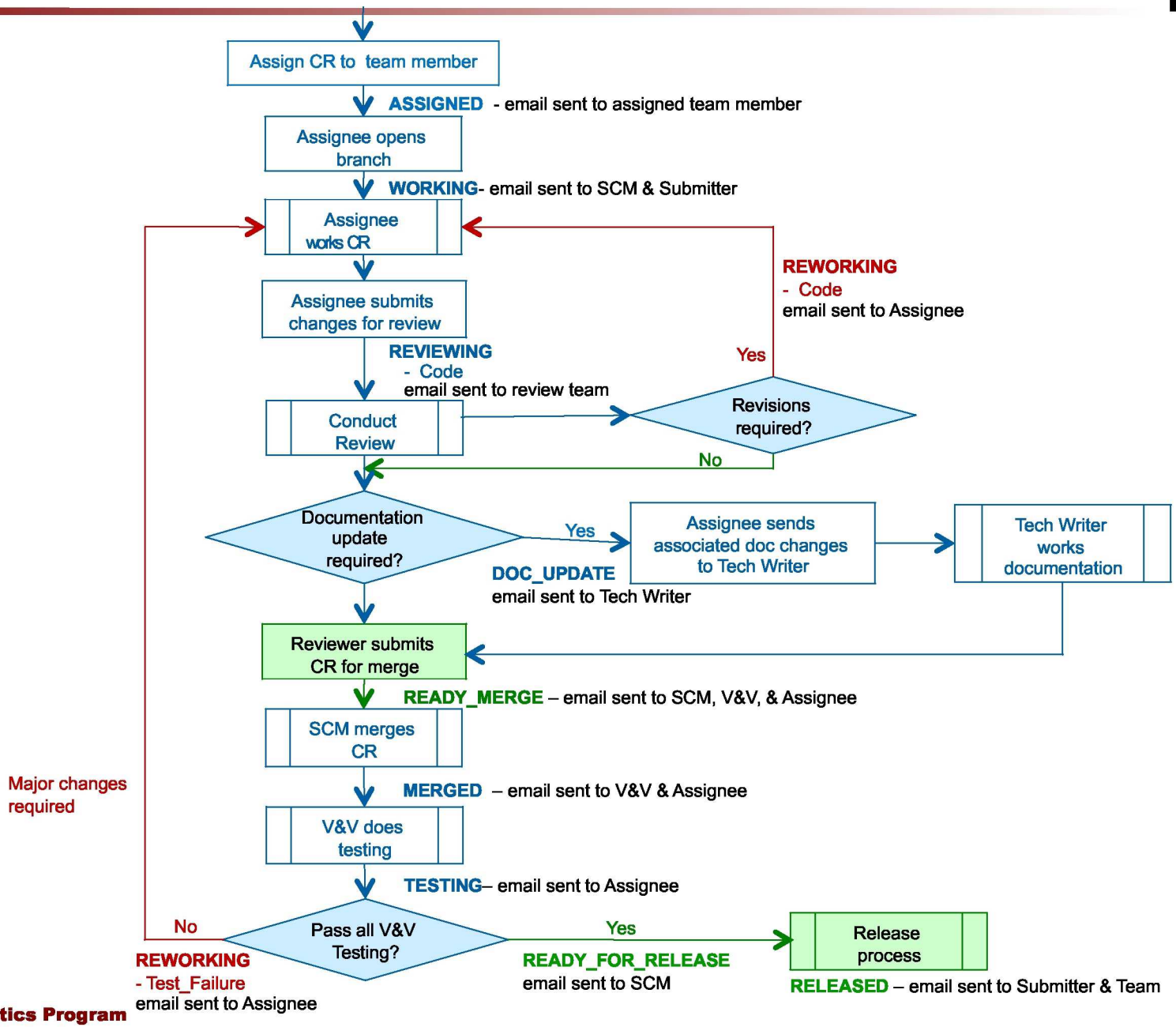
"An Open Source distribution, coupled with the opportunities provided by the Internet, would enable far greater dissemination than has ever been achieved before."

NASA Ames Open Source Software Website http://opensource.arc.nasa.gov/

OpenMDAO Life Cycle Process



OpenMDAO Life Cycle Process (cont'd)



Stochastic Design Optimization Code (SDOC)

Structures:

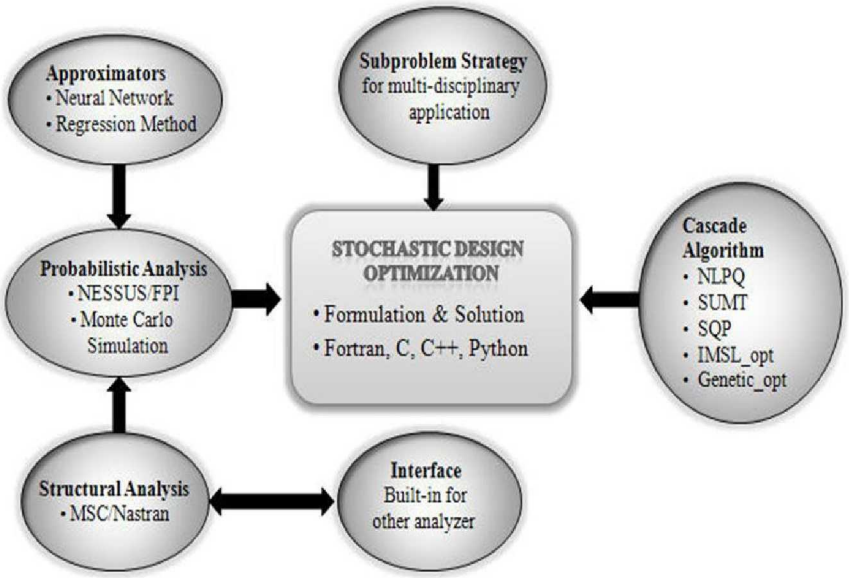
- Developing Stochastic Design Optimization Code (SDOC) & will be integrated using OpenMDAO
- Salient features: Structural design, Cascade algorithm, Subproblem strategy, Neural Network and Regression approximations.

Publications
(authors S. N. Patnaik, S. S. Pai, and R. M. Coroneos)

- "Reliability Based Design Optimization of an Airframe Component," The Journal of Aerospace Engineering (European), Proceedings of the Institution of Mechanical Engineers, (in Press).

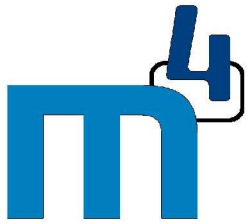
- "Reliability Based Design Optimization of a Composite Airframe Component," NASA Tech Brief, LEW-18497-1.

- "Optimization Test-bed CometBoards Extended into Stochastic Domain," AIAA SDM/Conference Paper, Palm Springs, CA, 4-7 May (2009).



Organization of SDOC

Integrated Multidisciplinary Optimization Objects



- Project Objectives:
  - Integrate a validated MDAO system into a Python-based, object-oriented framework (OpenMDAO).
    - The validated system integrated into OpenMDAO was High Fidelity Multidisciplinary Optimization (HFMDO), Build 1, developed under a separate NASA LaRC NRA
  - Demonstrate single-point and trade-study capabilities within OpenMDAO
  - Demonstrate adaptability of framework by integrating an additional common module (Mapping)
  - Provide feedback and lessons learned to OpenMDAO development team
- Integrated Modules were executed in Linux using Python 2.5 environment:
  - Geometry: utilizing GMAP
  - Aerodynamics: utilizing Panair
  - Propulsion: utilizing NPSS
  - Structures: utilizing NASTRAN
  - Mapping: utilizing M4 custom code
- Configuration analyzed was the High Speed Civil Transport (HSCT)
- Results:
  - Results between the original baseline single-point study and OpenMDAO study compare very well with a .35% maximum discrepancy
  - Results between the original sweep trade study and the OpenMDAO sweep trade study compare very well with a .60% maximum discrepancy
  - Results for the Mapping study are higher fidelity but require a longer run time
  - No major bugs or errors were discovered within the OpenMDAO framework
  - Minor one-time framework installation issues occurred
  - The M4 modules (Python-based) were compatible with OpenMDAO (Python-based).
  - Expertise with Python is a tremendous help